MCP361 Industrial Engineering Lab: Assignment 2

Due date: 9:00 AM August 7, 2024

- **Naming convention for files for this assignment is as follows**
  MCP361_Entry#_Assignment2_Problem1.py
  MCP361_Entry#_Assignment2_Problem2.py
  MCP361_Entry#_Assignment2_Problem3.py
  MCP361_Entry#_Assignment2_Problem4.py
  MCP361_Entry#_Assignment2.pdf

- **Submit a zip file to Moodle named as follows**
  MCP361_Entry#_Assignment2.zip

Remember the general guidelines for the assignments given at the start of the course.

****All plots for this assignment should have correct labeling and title, and they must be included in a PDF file, otherwise there will be penalty****

Q1) Generate n data points from the model, $y = a + bx +$ error, such that $a = 0.2$ and $b = 0.3$, with data points x being integers between 1 and 50 inclusive, and with errors drawn independently from the normal distribution with mean 0 and standard deviation 0.5.

[2 marks] Split 80% of the data into training set and rest 20% will be testing set. Code a python script to use **scikit-learn** machine learning library to train a linear regression model. Your code should print the slope and intercept of the learned LR model. The code should also output a **scatterplot** of the data and show the **fitted regression line**. Also, **draw** a bar plot such that x-axis shows all the x-data points from your testing set. For each x, you must draw two bars, one denoting the correct y-value and another denoting the predicted y-value.

[1 mark] Finally, your code should use the learned model to make predictions for the testing data set and use the predictions to compute: (i) Mean Absolute Error (ii) Mean Squared Error (iii) Root Mean Squared Error (iv) Coefficient of Determination (v) Explained Variance Score. You should **write** down the mathematical formulas for each of these five metrics in Overleaf and define each symbol used.

Q2) [1 mark] Code a python script to simulate n data points from the model, $y = a + bx +$ error such that $a = 0.2$ and $b = 0.3$, with data points x being integers between 1 and 5000 inclusive, and with errors drawn independently from the normal distribution with mean 0 and standard deviation 0.5. Split 80% of the data into training set and rest as testing set. Now use scikit-learn machine learning library to train a linear regression model. Then your code should use the learned model to make predictions for the testing data set and compute **the residual error** between predicted values and true values for the testing set.

[1 mark] The distribution of these errors should approximate a normal distribution according to your assumption. So, you must now check whether the residual errors you computed follow a normal distribution or not by **plotting** a histogram for the errors on the testing set. Also print the mean and standard deviation of that Normal distribution that best fits the error distribution. In the same plot as the histogram, plot the obtained Normal distribution showing how much it fits the histogram.

Q3) [3 marks] Code a python script to simulate n data points from the model, $y = a + bx +$ error, such that $a = 0.2$ and $b = 0.3$, with data points x being integers between 1 and 50 inclusive, and with errors drawn independently from the normal distribution with mean 0 and standard deviation 0.5. Now use ordinary least squares (OLS) function in **statsmodels** Python module to learn a linear regression model for the entire data set. Check if the true slope is:

i. within one standard error of the estimated slope.

ii. within two standard errors of the estimated slope.

Now do the entire process comprising data generation, model fitting, and slope checking 1000 times and report the fraction of times the true slope is

i. within one standard error of the estimated slope.

ii. within two standard errors of the estimated slope.

What value were you expecting for both the fractions and why? How close did you come to the expected values?

Q4) [2 marks] Code a python script to simulate n data points from the model, $y = a + bx + cx^2 +$ error, such that $a = 3, b = 8, c = 20$, with data points x being sampled at random 100 times from a uniform distribution on the range [0, 50], and with errors drawn independently from the normal distribution with mean 0 and standard deviation 3. Fit a straight-line using OLS estimator in statsmodels Python module.

However, you obviously expect a quadratic function to fit this data better, so apply the quadratic model to solve the regression problem using OLS estimator in statsmodels Python module. **Justify** using AIC values, which model is better out of the two? **Were** you able to recover the coefficients a, b, c using quadratic model fitting?